Package: pedtricks (via r-universe)

September 13, 2024

Title Visualize, Summarize and Simulate Data from Pedigrees

Version 0.4.3

Description Sensitivity and power analysis, for calculating statistics describing pedigrees from wild populations, and for visualizing pedigrees. This is a reboot of the methods developped by Morrissey and Wilson (2010) <doi:10.1111/j.1755-0998.2009.02817.x>

URL https://juliengamartin.github.io/pedtricks/

BugReports https://github.com/JulienGAMartin/pedtricks/issues

Language en-US

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

LazyData true

LazyLoad yes

Depends R (>= 4.0)

Imports ggplot2, dplyr, kinship2, tidyr, Matrix, mvtnorm, MCMCglmm, nadiv, genetics, methods, igraph

Repository https://juliengamartin.r-universe.dev

RemoteUrl https://github.com/juliengamartin/pedtricks

RemoteRef HEAD

RemoteSha a5ce75084f571c8973dd371c859ec8d3f0277500

Contents

`convert_ped` *Converts a pedigree with individuals specified as factors to a numeric pedigree*

Description

Some internal `pedtricks` modules require that pedigrees be specified only by numerical values, or including numerical values for missing data. This function provides the conversion to numeric but also back to factors if needed

Usage

```
convert_ped(type = "numeric", id, sire, dam, missingVal = NA, key = NULL)
```

Arguments

`type`	define how to convert the pedigree so "numeric" or "factor"
`id`	Individual identifiers - pass using `as.character()`
`sire`	Sire codes - pass `using as.character()`
`dam`	Dam codes - pass `using as.character()`
`missingVal`	the indicator that should be substituted for missing values
`key`	A dataframe, as produced by `convert_ped`, specifying factor codes for numeric values in id, sire, and dam

Value

`numericPedigree`	The factor pedigree in numeric form
`idKey`	A key to facilitate conversion back to the original identifiers

Examples

```
pedigree <- as.data.frame(matrix(c(
  "m1",   NA,     NA,
  "m2",   NA,     NA,
  "m3",   NA,     NA,
  "d4",   NA,     NA,
  "d5",   NA,     NA,
  "o6",   "m1",   "d4",
  "o7",   "m1",   "d4",
  "o8",   "m1",   "d4",
  "o9",   "m1",   "d4",
  "o10",  "m2",   "d5",
  "o11",  "m2",   "d5",
  "o12",  "m2",   "d5",
  "o13",  "m2",   "d5",
  "o14",  "m3",   "d5",
  "o15",  "m3",   "d5",
  "o16",  "m3",   "d5",
  "o17",  "m3",   "d5"
), 17, 3, byrow = TRUE))
names(pedigree) <- c("id", "dam", "sire")
for (x in 1:3) pedigree[, x] <- as.factor(pedigree[, x])

## make the test pedigree numeric with NAs denoted by -1
convert_ped(
  type = "numeric",
  id = as.character(pedigree[, 1]),
  dam = as.character(pedigree[, 2]),
  sire = as.character(pedigree[, 3]),
  missingVal = -1
)
```

`draw_ped` *Produce a graphical representation of a pedigree*

Description

Plots a pedigree, with options specific to considerations for pedigrees used to for quantitative genetic inference in natural populations. Pedigrees containing only those individuals that are informative with respect to (genetic) variation in an arbitrary trait can be plotted, potentially overlain on a complete pedigree. Functions also exist to plot various types of pedigree links associated with focal individuals.

Usage

```
draw_ped(
  Ped,
  cohorts = NULL,
  sex = NULL,
```

```
  dat = NULL,
  dots = "n",
  plotfull = "y",
  writeCohortLabels = "n",
  links = "all",
  sexInd = c(0, 1),
  dotSize = 0.001,
  dataDots = "n",
  dataDots.cex = 2,
  cohortLabs.cex = 1,
  retain = "informative",
  focal = NULL,
  sexColours = c("red", "blue"),
  ...
)
```

Arguments

`Ped`	An ordered pedigree with 3 columns: id, dam, sire
`cohorts`	An optional numeric vector of the same length as the pedigree designating, for example cohort affinities or birth years
`sex`	An optional numeric vector of the same length as the pedigree containing the sexes (may be unknown) of all individuals with entries in the pedigree. Defaults (modifiable with `sexInd`) are 0=male and 1=female
`dat`	An optional vector or data frame containing indicators of data availability. If dat contains only ones and zeros, then any individual with any entry of one will be considered as having data records. If data contains values other than ones and zeros, individuals in the pedigree with rows in data that contain at least one available record, i.e., one data record is not NA, will be treated as having data.
`dots`	If 'y', then a dot will be printed representing each individual in the pedigree. If sexes are available, dots will be colour coded by sex.
`plotfull`	To be used when dat is supplied. If 'y' (the default), individuals in the pedigree that are uninformative with respect to the available data have their pedigree links plotted in gray.
`writeCohortLabels`	To be used when cohorts is used. Will plot the cohort values on the left hand side of the pedigree image.
`links`	Default is 'all', other values are 'mums' to print only maternal pedigree links and 'dads' to print only paternal pedigree links.
`sexInd`	To be used with if sex is supplied and if the vector of sex specifiers differs from the default.
`dotSize`	Set the dot size bigger or smaller
`dataDots`	Will print dots over the dots denoting individuals, but denoting individuals with available data as indicated by dat.
`dataDots.cex`	controls the size of dataDots relative to dots.

cohortLabs.cex	controls the size of cohort labels.
retain	When those pedigree links only informative relative to phenotypic data availability are to be plotted, this controls whether or not a pruned pedigree based on phenotypic data is plotted (if set to "pruned"), or whether strictly only those informative pedigree links are plotted (the default)
focal	An optional list containing the id of an individual and the kinds of relatives of the focal individual to which to plot pedigree links. Available types are 'offspring','descendants','parents',,ancestors', and 'kin'.
sexColours	The colours that will be used to draw points and or lines associated with males and females.
...	Additional graphical parameters.

Value

output a plot of the pedigree, and does not return a value

Author(s)

Michael Morrissey <michael.morrissey@st-andrews.ac.uk>

See Also

fix_ped to prepare pedigrees that may not explicitly contain records for all individuals (i.e., where founding individuals may only appear in the dam or sire column).)

Examples

```
data(gryphons)
pedigree <- fix_ped(gryphons[, 1:3])

## draw the gryphon pedigree by pedigree depth
draw_ped(pedigree)

## draw the gryphon pedigree by cohort
draw_ped(pedigree,
  cohorts = gryphons$cohort, writeCohortLabels = "y",
  cohortLabs.cex = 1
)

## draw the gryphon pedigree by cohort with only maternal links
draw_ped(pedigree, cohorts = gryphons$cohort, links = "mums")

## draw the gryphon pedigree by cohort with colour only for those
## indiduals that are informative relative to the quantitative
## genetics of a hypothetical trait only measured for individuals
## in the last two cohorts, emphasize the phenotyped individuals
## with large black dots, and all other individuals with dots
## colour coded by sex:
dataAvailability <- (gryphons$cohort >= (max(gryphons$cohort) - 1)) + 0
```

```
draw_ped(pedigree,
  cohorts = gryphons$cohort, sex = gryphons$sex,
  dots = "y", dat = dataAvailability, writeCohortLabels = "y", dataDots = "y"
)
```

---

draw_pedA *Produce a graphical representation of the relatedness matrix of a pedigree*

Description

Creates the object needed to plot a pedigree's numerator relatedness matrix given a few different choices for ordering. The resulting image for a pedigree of size n can be visualized as a n x n grid of colored squares based on values of the numerator relatedness matrix.

Usage

```
draw_pedA(
  pedigree,
  order = c("original", "generation", "user"),
  grp = NULL,
  ...
)
```

Arguments

pedigree	A data.frame of a pedigree with 3 columns: id, dam, sire
order	A character expression for how the pedigree should be ordered for visualization. See Details.
grp	A character expression for the column name in pedigree indicating how to order the pedigree for visualization.
...	Additional plotting arguments passed to image.

Value

A list of class "trellis".

Author(s)

<mew0099@auburn.edu>

Examples

```
data(gryphons)
pedigree <- fix_ped(gryphons[, 1:3])

## draw the gryphon pedigree
draw_pedA(pedigree, order = "original")

## draw the gryphon pedigree by function assigned generation
(Agen <- draw_pedA(pedigree, order = "generation"))



## draw the gryphon pedigree by cohort in the dataset
## add cohort back from original data
pedigree$cohort <- NA
pedigree$cohort[match(gryphons$id, pedigree[, 1])] <- gryphons$cohort
(Achrt <- draw_pedA(pedigree, order = "user", grp = "cohort"))

## show two images of the same pedigree in different orders
### (i.e., plotting multiple trellis objects in the same figure)
plot(Agen,
  position = c(xmin = 0, ymin = 0, xmax = 0.45, ymax = 1),
  more = TRUE
)
plot(Achrt, position = c(xmin = 0.55, ymin = 0, xmax = 1, ymax = 1))
```

`fix_ped` *Manipulating pedigrees to prepare them for requirements of subsequent analyses Prepares a pedigree to conform with requirements of many softwares used in quantitative genetic analysis, as well as for many of the functions in pedtricks.*

Description

Manipulating pedigrees to prepare them for requirements of subsequent analyses Prepares a pedigree to conform with requirements of many softwares used in quantitative genetic analysis, as well as for many of the functions in pedtricks.

Usage

```
fix_ped(ped, dat = NULL)
```

Arguments

`ped`	An ordered pedigree with 3 columns: id, dam, sire
`dat`	An optional data frame, the same length as the pedigree

Value

Returns a pedigree in which all individuals that exist in the dam and sire columns are represented by their own record lines, occurring before the records of their first offspring. If data are supplied, then fix_ped will return a dataframe, the first three columns are the 'fixed' pedigree, and the following columns of which contain appropriately reordered data.

Examples

```
##  a valid pedigree, i.e., no loops, no bisexuality, etc.,
## but where not all parents have a record line, and where
## parents do not necessarily occur before their offspring:
pedigree <- as.data.frame(matrix(c(
  10, 1, 2,
  11, 1, 2,
  12, 1, 3,
  13, 1, 3,
  14, 4, 5,
  15, 6, 7,
  4, NA, NA,
  5, NA, NA,
  6, NA, NA,
  7, NA, NA
), 10, 3, byrow = TRUE))
names(pedigree) <- c("id", "dam", "sire")
pedigree
fixed_pedigree <- fix_ped(ped = pedigree)
fixed_pedigree
```

---

genome_sim *A function to simulate QTL and/or SNP data.*

Description

Simulates a chromosome of arbitrary length with arbitrary numbers, types, and spacings of genetic loci over arbitrary pedigrees.

Usage

```
genome_sim(
  pedigree,
  founders = NULL,
  positions = NULL,
  initHe = NULL,
  mutationType = NULL,
  mutationRate = NULL,
  phenotyped = NULL,
  founderHaplotypes = NULL,
```

```
  genotyped = NULL,
  returnG = "n",
  initFreqs = NULL
)
```

Arguments

pedigree	A pedigree
founders	A vector of indicator variables denoting founder status (1=founder, 0=non-founder)
positions	Genome locations in cM for markers
initHe	Initial levels of expected heterozygosity
mutationType	A vector of locus types - see details
mutationRate	A vector of mutation rates
phenotyped	A vector of IDs of those individuals for which to return phenotypic data
founderHaplotypes	A matrix or dataframe containing founder haplotypes
genotyped	A vector of IDs of those individuals for which to return genotypic data
returnG	If 'y' then genotypic data for all loci (including cIAM loci) will be returned.
initFreqs	A list of allele frequencies for all loci. If initFreqs is specified, it will override information from initHe. extractA from package MasterBayes can be used to obtain obtain initFreqs form a sample of genotypes. For cIAM loci, allele names in initFreqs should be allelic substitution effects.

Details

Valid mutation types are Micro', Dom', dIAM' and cIAM', for microsatellite, dominant (AFLP), discrete infinite alleles mutation model loci (SNPs), and continuous infinite alleles mutation model loci (polymorphisms effecting phenotypic variation). cIAM loci have mutational allelic substitution effects taken drawn from a normal distribution with mean 0 and variance 1.

Value

Phenotypes	A vector of phenotypes. Calculated as the sum of all allelic effects. Scaling is currently left to be done post-hoc.
MarkerData	A vector of marker genotypes, i.e. alleles at all loci except those designated 'cIAM'

See Also

phen_sim, micro_sim

Examples

```
testData <- as.data.frame(matrix(
  c(
    1,      NA,     NA,     1,      1,      1,      2,      2,
    2,      NA,     NA,     1,      1,      1,      2,      2,
    3,      NA,     NA,     1,      1,      1,      2,      2,
    4,      NA,     NA,     1,      0,      1,      2,      2,
    5,      NA,     NA,     1,      0,      1,      2,      2,
    6,      1,      4,      0,      -1,     2,      3,      3,
    7,      1,      4,      0,      -1,     2,      3,      3,
    8,      1,      4,      0,      -1,     2,      3,      3,
    9,      1,      4,      0,      -1,     2,      3,      3,
    10,     2,      5,      0,      -1,     2,      3,      3,
    11,     2,      5,      0,      -1,     2,      3,      3,
    12,     2,      5,      0,      -1,     2,      3,      3,
    13,     2,      5,      0,      -1,     2,      3,      3,
    14,     3,      5,      0,      -1,     2,      3,      3,
    15,     3,      5,      0,      -1,     2,      3,      3,
    16,     3,      5,      0,      -1,     2,      3,      3,
    17,     3,      5,      0,      -1,     2,      3,      3
  ),
  17, 8,
  byrow = TRUE
))

names(testData) <- c(
  "id", "dam", "sire", "founder", "sex",
  "cohort", "first", "last"
)
pedigree <- as.data.frame(cbind(
  testData$id, testData$dam,
  testData$sire
))
for (x in 1:3) pedigree[, x] <- as.factor(pedigree[, x])
names(pedigree) <- c("id", "dam", "sire")
pedigree

## make up some microsatellite and gene allele frquencies:
sampleGenotypes <- as.data.frame(matrix(c(
  1, 2, -1.32, 0.21, 2, 1, 0.21, 0.21
), 2, 4, byrow = TRUE))
testFreqs <- extractA(sampleGenotypes)

## note that alleles at the gene locus are given as their
## allelic substitution effects:
testFreqs

## simulate data for these indivdiuals based on a single QTL
## with two equally alleles with balanced frequencies in the
## founders, linked (2 cM) to a highly  polymorphic microsatellite:
genome_sim(
  pedigree = pedigree, founders = testData$founder, positions = c(0, 2),
```

```
  mutationType = c("Micro", "cIAM"), mutationRate = c(0, 0),
  initFreqs = testFreqs, returnG = "y"
)
## since we specified returnG='y', we can check that
## the phenotypes add up to the
## allelic substitution effects for the second locus.
```

---

ggpedigree *ggpedigree: Plotting tool for simple and complex pedigrees.*

Description

This function plots simple and complex pedigrees, with options specific to the types of pedigrees used for quantitative genetic inference in natural populations. This function is flexible to missing parents and can be customized to visualise specific cohorts, sexes, and/or phenotype availability. Pedigree layout is optimized using a Sugiyama algorithm. For simpler pedigrees, visualisation may be improved by specifying `spread_x_coordinates = FALSE`.

Usage

```
ggpedigree(
  .data,
  ids,
  mothers,
  fathers,
  cohort,
  sex,
  pheno,
  sex_code = NULL,
  id_labels = FALSE,
  remove_singletons = TRUE,
  plot_unknown_cohort = FALSE,
  spread_x_coordinates = TRUE,
  print_cohort_labels = TRUE,
  return_plot_tables = FALSE,
  line_col_mother = "#E41A1C",
  line_col_father = "#377EB8",
  line_col_no_pheno = "#aaaaaa",
  line_alpha = 0.3,
  point_size = 1,
  point_colour = "black",
  point_alpha = 1
)
```

Arguments

.data	a data frame object with all the pedigree information
ids	a column of .data of individual identifiers
mothers	A column of .data of mothers corresponding to ids. Missing values are 0 or NA.
fathers	A column of .data of fathers corresponding to ids. Missing values are 0 or NA.
cohort	integer. Default NULL. A optional column of .data assigning a cohort to each id. If NULL, then kinship2::kindepth is used to assign cohorts to ids.
sex	integer or character. Default NULL. An optional column of .data assigning a sex to each id. When using this option, sex_code must be specified. Any values not matching values in sex_code will be treated as unknown sex.
pheno	integer or character. Default NULL. An optional column of .data assigning a phenotype to each id. Links originating from parents that have NA values for this argument will be plotted with a grey line, unless otherwise specified in line_col_no_pheno.
sex_code	Default NULL. A vector of length 2, indicating the value used in sex for females and males respectively. Females are plotted as circles, males as squares, and unknown values as triangles.
id_labels	logical. Default FALSE. Print the ids on the pedigree plot.
remove_singletons	logical. Default TRUE. Remove ids with no relatives i.e., no offspring or parents assigned.
plot_unknown_cohort	logical. Default FALSE. Plots ids of unknown cohorts. These are plotted in an "Unknown" cohort at the top of the pedigree. Be aware that any mothers and fathers of these individuals will be plotted below them.
spread_x_coordinates	logical. Default TRUE. Evenly spreads the x-axis (horizontal) distribution of points within each cohort. If FALSE, this will plot the direct outcome of igraph::layout_with_sugiyam the FALSE option is only recommended for small pedigrees and/or less connected pedigrees.
print_cohort_labels	logical. Default TRUE. Prints cohort levels on the left hand side of plot.
return_plot_tables	logical. Default FALSE. Returns an object with the line and point data used for the plot, but the plot will not be generated
line_col_mother	Default = "#E41A1C". Line colour for maternal links.
line_col_father	Default = "#377EB8". Line colour for paternal links.
line_col_no_pheno	Default = "#aaaaaa". Line colour for parents with NA values in pheno.
line_alpha	Default = 0.3. Line alpha (transparency) value for maternal and paternal links.
point_size	Default = 1. Point size for ids.
point_colour	Default = "black". Point colour for ids.
point_alpha	Default = 1. Point alpha for ids.

Value

output a ggplot object or a list of tables if `return_plot_tables = TRUE`

Examples

```
data(gryphons)
pedigree <- fix_ped(gryphons[, 1:3])

## draw the gryphon pedigree by pedigree depth
ggpedigree(pedigree)


# specifying the column names for id, mother and father
ggpedigree(pedigree, id, dam, sire)

# with cohort and sex
ggpedigree(gryphons, cohort = cohort, sex = sex, sex_code = c(1, 0))

#' with cohort, sex, and pheno
gryphons$pheno <- 1
gryphons$pheno[sample(length(gryphons$pheno), 1000)] <- NA
ggpedigree(gryphons, cohort = cohort, sex = sex, sex_code = c(1, 0), pheno = pheno)
```

---

gryphons *Example dataset for pedtricks examples and tutorial*

Description

This contains pedigree and life history data of a fictional population. The data are relevant to power and sensitivity analyses for quantitative genetic studies of natural populations.

Usage

```
gryphons
```

Format

An object of class `data.frame` with 4918 rows and 9 columns.

Details

id what

dam what

sire what

sex what

cohort what

fatherErrorProb what

fatherSampledProb what

firstReproCohort what

lastReproCohort what

`micro_sim` *Simulates microsatellite data across a pedigree.*

Description

Uses a pedigree with parents identified for all non-founding individuals and simulates microsatellite genotypes

Usage

```
micro_sim(
  pedigree,
  genFreqs = NULL,
  genotypesSample = NULL,
  knownGenotypes = NULL,
  records = NULL,
  eRate1 = 0,
  eRate2 = 0,
  eRate3 = 0
)
```

Arguments

`pedigree`	A pedigree
`genFreqs`	(optional) A list of allele frequencies, can be produced with `extractA`
`genotypesSample`	(required if `genFreqs` is not supplied) a sample of genotypes from which to estimate population allele frequencies
`knownGenotypes`	(not yet implemented) a data frame of genotypes for (potentially a subset) of founder individuals
`records`	Record availability, see details.
`eRate1`	The rate of genotypic substitution errors, i.e., when a true genotype at a given locus is replaced by a pair of alleles selected at random based on the population allele frequencies
`eRate2`	The rate of allelic substitution errors, i.e. when an allele is erroneously replaced at a given locus by an allele chosen at random based on the population allele frequencies
`eRate3`	The rate of large allele dropouts, simulated by setting the value of the larger allele at a locus to the value of the smaller allele

Details

Error rates and data availability rates can be specified as either (1) single values to be applied to all individuals and all loci, (2) as a vector the same length as the number of loci, representing locus-specific rates to be applied uniformly to all individuals, or (3) as data frames with rows for each individual and columns for each locus. In the third option, observed patterns of data availability can be simulated by supplying 0s and 1s for missing and available individual genotypes, respectively.

Value

`trueGenotypes` A data frame of true genotypes

`observedGenotypes`
A data frame of plausible observed genotypes, given specified patterns of missingness and errors.

See Also

phen_sim, genome_sim

Examples

```
pedigree <- as.data.frame(matrix(c(
  "m1",   NA,     NA,
  "m2",   NA,     NA,
  "m3",   NA,     NA,
  "d4",   NA,     NA,
  "d5",   NA,     NA,
  "o6",   "m1",   "d4",
  "o7",   "m1",   "d4",
  "o8",   "m1",   "d4",
  "o9",   "m1",   "d4",
  "o10",  "m2",   "d5",
  "o11",  "m2",   "d5",
  "o12",  "m2",   "d5",
  "o13",  "m2",   "d5",
  "o14",  "m3",   "d5",
  "o15",  "m3",   "d5",
  "o16",  "m3",   "d5",
  "o17",  "m3",   "d5"
), 17, 3, byrow = TRUE))
names(pedigree) <- c("id", "dam", "sire")
for (x in 1:3) pedigree[, x] <- as.factor(pedigree[, x])

## some sample genotypes, very simple, two markers with He = 0.5
sampleGenotypes <- as.data.frame(matrix(c(
  1, 2, 1, 2, 2, 1, 2, 1
), 2, 4, byrow = TRUE))
## locus names
names(sampleGenotypes) <- c("loc1a", "loc1b", "loc2a", "loc2b")

## simulate some genotypes
micro_sim(pedigree = pedigree, genotypesSample = sampleGenotypes)
```

`ped_stats` *Calculates a range of statistics of pedigrees*

Description

Statistics are those that will hopefully be useful for describing pedigrees to be used in quantitative genetic analyses of natural populations. This module will be most useful when cohort affinities for all individuals can be provided. All outputs are produced in a numerical form as well as in graphical summaries.

Usage

```
ped_stats(
  Ped,
  cohorts = NULL,
  dat = NULL,
  retain = "informative",
  includeA = TRUE,
  lowMem = FALSE
)
```

Arguments

`Ped`	A pedigree
`cohorts`	(Optional) Cohort affinities for members of the pedigree
`dat`	(Optional) Available data based upon which the pedigree can be pruned for just informative individuals
`retain`	The default value ('informative') results in pedigree being pruned to only those individuals who's records contribute to estimation of quantitative genetic parameters with respect to the available data specified in `dat`. Otherwise, specifying a value of 'ancestors' will result in the inclusion of all ancestors of phenotyped individuals.
`includeA`	If TRUE, additive genetic relatedness matrix is returned.
`lowMem`	If TRUE, then stats based on calculation of A are not performed.

Value

`totalMaternities`	Total number of maternities defined by the pedigree.
`totalPaternities`	Total number of paternities defined by the pedigree.
`totalFullSibs`	Total number of pair-wise full sib relationships defined by the pedigree.

`totalMaternalSibs`
: Total number of pair-wise maternal sib relationships defined by the pedigree. To get the number of maternal half sibs, subtract totalFullSibs.

`totalPaternalSibs`
: Total number of pair-wise paternal sib relationships defined by the pedigree. To get the number of paternal half sibs, subtract totalFullSibs.

`totalMaternalGrandmothers`
: Total number of maternal grandmothers defined by the pedigree.

`totalMaternalGrandfathers`
: Total number of maternal grandfathers defined by the pedigree.

`totalPaternalGrandmothers`
: Total number of paternal grandmothers defined by the pedigree.

`totalPaternalGrandfathers`
: Total number of paternal grandfathers defined by the pedigree.

`pedigreeDepth`
: The pedigree depth, i.e. maximum number of ancestral generations, for each individual.

`inbreedingCoefficients`
: Individual inbreeding coefficients

`maternalSibships`
: Sibship size of each individual appearing the the dam column of the pedigree.

`paternalSibships`
: Sibship size of each individual appearing the the sire column of the pedigree.

`cumulativeRelatedness`
: Proportion of pair-wise relatedness values less than values ranging from 0 to 1.

`relatednessCategories`
: Discretized distribution of relatedness.

`analyzedPedigree`
: Returns the pedigree.

`sampleSizesByCohort`
: (Optional) Number of individuals belonging to each cohort.

`maternitiesByCohort`
: (Optional) Number of assigned maternities by offspring cohort.

`paternitiesByCohort`
: (Optional) Number of assigned paternities by offspring cohort.

`fullSibsByCohort`
: (Optional) Number of pair-wise full sib relationships by cohort - note the sum of these need not be equal to totalFullSibs in pedigrees of long-lived organisms.

`maternalSibsByCohort`
: (Optional) Number of pair-wise maternal sib relationships by cohort - note the sum of these need not be equal to totalMaternalSibs in pedigrees of long-lived organisms.

`paternalSibsByCohort`
: (Optional) Number of pair-wise paternal sib relationships by cohort - note the sum of these need not be equal to totalPaternalSibs in pedigrees of long-lived organisms.

`maternalGrandmothersByCohort`
: (Optional) Numbers of maternal grandmother assignments by offspring cohort.

`maternalGrandfathersByCohort`
: (Optional) Numbers of maternal grandmother assignments by offspring cohort.

`paternalGrandmothersByCohort`
: (Optional) Numbers of paternal grandfather assignments by offspring cohort.

`paternalGrandfathersByCohort`
: (Optional) Numbers of paternal grandfather assignments by offspring cohort.

`cumulativePedigreeDepth`
: (Optional) Distributions of pedigree depth by cohort.

`meanRelatednessAmongCohorts`
: (Optional) Mean relatedness among cohorts.

`cohorts`
: (Optional) Returns cohort designations.

Graphical summaries of a number of these summary statistics are printed to the console when `graphicalReports=='y'`.

Examples

```
data(gryphons)
pedigree <- gryphons[, 1:3]

gryphons_ped_stats <- ped_stats(pedigree,
  cohorts = gryphons$cohort
)

gryphons_ped_stats$totalMaternities
gryphons_ped_stats$paternitiesByCohort

summary(gryphons_ped_stats)

plot(gryphons_ped_stats)
```

`phen_sim` *A function to simulated phenotypic data*

Description

Simulates phenotypic data across arbitrary pedigrees. `phen_sim` simulate direct, maternal and paternal genetic and environmental effects for an arbitrary number of traits with arbitrary patterns of missing data.

Usage

```
phen_sim(
  pedigree,
  traits = 1,
  randomA = NULL,
  randomE = NULL,
  parentalA = NULL,
  parentalE = NULL,
  sampled = NULL,
  records = NULL,
  returnAllEffects = FALSE,
  verbose = TRUE
)
```

Arguments

pedigree	A pedigree
traits	The number of traits for which data should be simulated.
randomA	An additive genetic covariance matrix, with dimensions a multiple of traits - see details
randomE	An additive environmental covariance matrix, with dimensions a multiple of traits - see details
parentalA	A vector indicating which effects in randomA (if any) to treat as parental effects
parentalE	A vector indicating which effects in randomE (if any) to treat as parental effects
sampled	A vector indicating which individuals are sampled
records	A single value, array of matrix specifying data record availability - see details
returnAllEffects	If TRUE then all individual breeding values and environmental effects are returned
verbose	If TRUE provide a progress bar and messages, Default: TRUE

Details

randomA and randomE are square matrices with dimension equal to the sum of the number direct and indirect effects. This must be a multiple of the number of traits, i.e. if an indirect effect is to be simulated for only one of multiple traits, those traits with no indirect effect should be included with (co)variances of zero.

parentalA and parentalE are optional vectors of characters indicating which trait positions in randomA and randomE are to be treated as indirect effects, and which effects to treat as maternal or paternal. Valid values are 'd', 'm', and 'p', for direct, maternal indirect and paternal indirect effects, respectively.

records can be specified either (1) as a single value to be applied to all individuals and traits, (2) as a vector the same length as the number of traits, representing trait-specific rates to be applied uniformly to all individuals, or (3) as data frames with rows for each individual and columns for each trait. In the third option, observed patterns of data availability can be simulated by supplying 0s and 1s for missing and available individual genotypes, respectively.

Value

phenotypes	A dataframe containing phenotypes for all individuals specified to have records.
allEffects	(optional) A dataframe with all direct and indirect genetic and environmental effects.

See Also

micro_sim, genome_sim

Examples

```
## make up a pedigree
id <- c("a1", "a2", "a3", "a4", "a5", "a6", "a7", "a8", "a9")
dam <- c(NA, NA, NA, "a1", "a1", "a1", "a4", "a4", "a4")
sire <- c(NA, NA, NA, "a2", "a2", "a2", "a5", "a6", "a6")
pedigree <- as.data.frame(cbind(id, sire, dam))

traits <- 2
## no correlations
randomA <- diag(4)
randomE <- diag(4)
parentalA <- c("d", "d", "m", "m")
parentalE <- c("d", "d", "m", "m")

## generate phenoypic data based on this architecture
phen_sim(
  pedigree = pedigree, traits = 2, randomA = randomA, randomE = randomE,
  parentalA = parentalA, parentalE = parentalE
)

## let's do it again but see how the phenotypes were composed
phen_sim(
  pedigree = pedigree, traits = 2, randomA = randomA, randomE = randomE,
  parentalA = parentalA, parentalE = parentalE, returnAllEffects = TRUE
)
```

---

plot.ped_stats *Plot output from ped_stats*

Description

Generates a manageable summary of pedigree-wide statistics reported by ped_stats, either for a single pedigree or for a comparison between pedigrees

Usage

```
## S3 method for class 'ped_stats'
plot(x, lowMem = FALSE, grContrast = FALSE, ...)
```

Arguments

x	An object of class ped_stats generated by ped_stats
lowMem	If TRUE, then stats based on calculation of A are not performed.
grContrast	If TRUE, then uglier shades of red and blue are used to denote male and female statistics in graphical reports, but these colours provide better contrast in greyscale.
...	extra arguments

Value

Returns a table of numbers of records, maternities, paternities, pairwise sibship relationships, numbers of different classes of grand-parental relationships, pedigree depth, number of founders, mean sibship sizes, simple statistics of numbers of inbred and non-inbred individuals, and proportions of pairwise relationship coefficients equal to or greater than several thresholds.

---

summary.ped_stats *Post-processes output from ped_stats*

Description

Generates a manageable summary of pedigree-wide statistics reported by ped_stats, either for a single pedigree or for a comparison between pedigrees

Usage

```
## S3 method for class 'ped_stats'
summary(object, ...)
```

Arguments

object	An object of class ped_stats generated by ped_stats
...	extra arguments

Value

Returns a table of numbers of records, maternities, paternities, pairwise sibship relationships, numbers of different classes of grand-parental relationships, pedigree depth, number of founders, mean sibship sizes, simple statistics of numbers of inbred and non-inbred individuals, and proportions of pairwise relationship coefficients equal to or greater than several thresholds.

`WarblerG` *Seychelles Warbler Genotypes*

Description

Genotype data collected by David Richardson from Cousin Island in 1999.

Usage

```
WarblerG
```

Format

An object of class `data.frame` with 307 rows and 29 columns.

References

Richardson *et.al.* (2001) Molecular Ecology 10 2263-2273 Hadfield J.D. *et al* (2006) Molecular Ecology 15 3715-31

Index